

Java Chapter 10: Files

Contents

Java Chapter 10: Files	1
Do: Online Tutorials.....	1
Files	1
Tutorial 8.1 – Reading a Text File	2
Tutorial 8.2 – Writing a Text File	3
Tutorial 8.3 – Reading a Text File Line by Line	4
Tutorial 8.4 – Appending to a Text File.....	6
Tutorial 8.5 – Writing Numbers to a Text File	8
Tutorial 8.6 – Read Numbers from a Text File	9
Tutorial 8.7 – Reading and Writing to a Text File	10
Assignment Submission.....	12

Time required: 60 minutes

Do: Online Tutorials

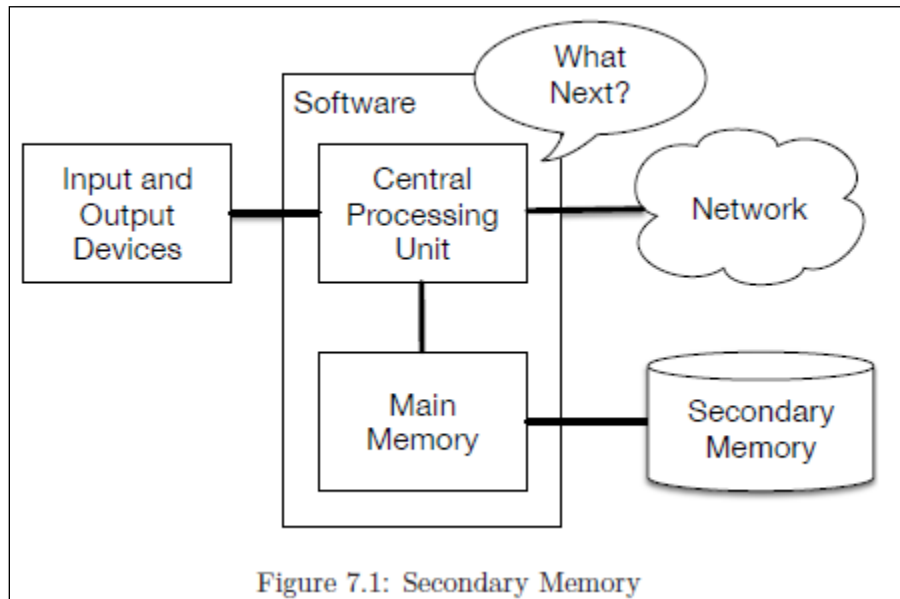
- [Java Files](#)
- [Java Create and Write to Files](#)
- [Java Read Files](#)
- [Java Delete Files](#)

Files

Most of the programs we have seen so far are transient in the sense that they run for a short time and produce some output. When they end, their data disappears. If you run the program again, it starts with a clean slate.

Other programs are persistent: they run for a long time (or all the time); they keep at least some of their data in permanent storage (a hard drive, for example); and if they shut down and restart, they pick up where they left off.

Examples of persistent programs are operating systems, which run pretty much whenever a computer is on, and web servers, which run all the time, waiting for requests to come in on the network.



Secondary memory is not erased when the power is turned off. Or in the case of a USB flash drive, the data we write from our programs can be removed from the system and transported to another system.

Tutorial 8.1 – Reading a Text File

In the same folder, create a txt file named **example.txt** Include the following text.

```
Hello  
This is a sample text file.  
Bye!
```

Create and test the program as listed. Name it **file_read.py**

```

1  """
2      Name: file_read.py
3      Author:
4      Created:
5      Purpose: Read and display a text file
6  """
7
8
9  def main():
10
11      # Open a file in the same folder as the program
12      # We create an object named text_file
13      # This is called a file handle
14      text_file = open("example.txt", "r")
15
16      # Read the entire contents of the file into a string
17      file_contents = text_file.read()
18
19      # Close the file handle
20      text_file.close()
21
22      # Print the string
23      print(file_contents)
24
25
26  # If a standalone program, call the main function
27  # Else, use as a module
28  if __name__ == "__main__":
29      main()

```

Tutorial 8.2 – Writing a Text File

Create and test the following Python program called **file_rockstars_write.py**

This program uses try: except: to provide error handling. If there was an exception, inform the user.

Let the user know the operation was successful by including a print statement in the try block.

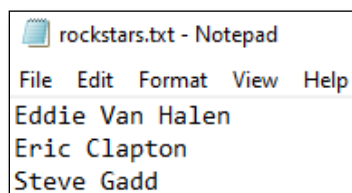
```

1  """
2      Name: file_rockstars_write.py
3      Author:
4      Created:
5      Purpose: Write 3 lines of data to a text file
6  """
7
8
9  def main():
10
11      # Catch any exceptions
12      try:
13          # Create a file handle
14          # for writing to rockstars.txt
15          with open("rockstars.txt", "w") as rock_file:
16
17              # Write the names of three rock stars to the file
18              # Substitute your favorites
19              # \n is an escape character creating a new line
20              rock_file.write(f"Eddie Van Halen\n")
21              rock_file.write(f"Eric Clapton\n")
22              rock_file.write(f"Steve Gadd\n")
23              print(f"File written successfully.")
24
25      # Let the user know there was an exception
26      except:
27          print(f"The file was not written.")
28
29
30 # Call the main function
31 if __name__ == "__main__":
32     main()

```

Open **rockstars.txt** to check your work.

Example run:



Tutorial 8.3 – Reading a Text File Line by Line

Create a new program called **file_rockstars_read.py** This program reads each line into a separate string.

```

1  """
2      Name: file_rockstars_read.py
3      Author:
4      Created:
5      Purpose: Read and display a text file line by line
6  """
7
8
9  def main():
10
11      # Catch any exceptions in the program
12      try:
13          # Open a file in the same folder as the program
14          with open('rockstars.txt', 'r') as text_file:
15
16              # Read each line into a separate string
17              line1 = text_file.readline()
18              line2 = text_file.readline()
19              line3 = text_file.readline()
20
21              # Print the strings
22              print(line1)
23              print(line2)
24              print(line3)
25              print('The file was successfully read.')
26
27          # Let the user know if there was an exception
28      except:
29          print('There was a problem reading the file.')
30
31
32  # Call the main function
33  if __name__ == '__main__':
34      main()

```

Because of the \n newline character we added when we wrote the file, there is a space between each line when we print out the strings.

```

Eddie Van Halen
Eric Clapton
Steve Gadd
The file was successfully read.

```

```

1  """
2      Name: file_rockstars_read_rstrip.py
3      Author:
4      Created:
5      Purpose: Read and display a text file line by line
6  """
7
8
9  def main():
10
11      # Catch any exceptions in the program
12      try:
13          # Open a file in the same folder as the program
14          with open('rockstars.txt', 'r') as text_file:
15
16              # Read each line into a separate string
17              line1 = text_file.readline()
18              line2 = text_file.readline()
19              line3 = text_file.readline()
20
21              # Strip \n from each string with the rstrip function
22              line1 = line1.rstrip('\n')
23              line2 = line2.rstrip('\n')
24              line3 = line3.rstrip('\n')
25
26              # Print the strings
27              print(line1)
28              print(line2)
29              print(line3)
30
31              print('The file was successfully read.')
32
33              # Let the user know if there was an exception
34          except:
35              print('There was a problem reading the file.')
36
37
38      # Call the main function
39      if __name__ == '__main__':
40          main()

```

The text file displayed without \n extra new line characters.

```

Eddie Van Halen
Eric Clapton
Steve Gadd
The file was successfully read.

```

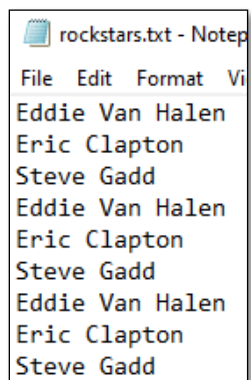
Tutorial 8.4 – Appending to a Text File

'a'	Opens a file for appending at the end of the file without erasing the contents. Creates a new file if it does not exist.
-----	---

Open **file_rockstars_write.py**. The only change needed is to change the 'w' to an 'a'. Run the program a couple times. Open the text file to make sure the program worked.

```
1  """
2      Name: file_rockstars_append.py
3      Author:
4      Created:
5      Purpose: Append 3 lines of data to a text file
6  """
7
8
9  def main():
10
11      # Catch any exceptions
12      try:
13          # Open a file for appending in the same folder named rockstars.txt
14          with open('rockstars.txt', 'a') as rock_file:
15
16              # Append the names of three rock stars to the file
17              # Substitute your favorites
18              # \n is an escape character creating a new line
19              rock_file.write('Eddie Van Halen' + '\n')
20              rock_file.write('Eric Clapton' + '\n')
21              rock_file.write('Stevie Nicks' + '\n')
22
23              print('File written successfully')
24
25      # Let the user know there was an exception
26      except:
27          print('Something went wrong and caused an exception')
28
29
30  # Call the main function.
31  main()
```

Example run:



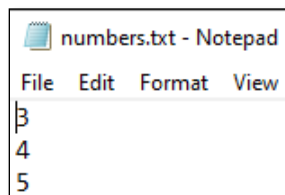
```
rockstars.txt - Notepad
File Edit Format Vi
Eddie Van Halen
Eric Clapton
Steve Gadd
Eddie Van Halen
Eric Clapton
Steve Gadd
Eddie Van Halen
Eric Clapton
Steve Gadd
```

Tutorial 8.5 – Writing Numbers to a Text File

Numbers must be converted to a string before they can be written to a text file. We will use Fstrings to convert the numbers to strings.

```
1  """
2      Name: file_numbers_write.py
3      Author:
4      Created:
5      Purpose: Numbers must be converted to strings before they
6                are written to a text file.
7  """
8  # Constant for filename
9  FILE_NAME = 'numbers.txt'
10
11
12 def main():
13
14     # Catch any exceptions
15     try:
16         # Open a file for writing
17         with open(FILE_NAME, 'w') as number_file:
18
19             # Get three numbers from the user.
20             number1 = int(input('Enter a whole number: '))
21             number2 = int(input('Enter another whole number: '))
22             number3 = int(input('Enter another whole number: '))
23
24             # Write the numbers to the file using Fstrings
25             number_file.write(f'{number1}\n')
26             number_file.write(f'{number2}\n')
27             number_file.write(f'{number3}\n')
28
29             # Let the user know it worked
30             print('Data was written to numbers.txt')
31
32     # Let the user know there was trouble
33     except:
34         print('There was trouble writing to the file.')
35
36
37 # Call the main function.
38 if __name__ == '__main__':
39     main()
```

Example run:



Tutorial 8.6 – Read Numbers from a Text File

Numbers stored as text in a text file must be converted from string to numbers before they can be used in calculations.

```
1  """
2      Name: file_read_numbers.py
3      Author:
4      Created:
5      Purpose: Numbers must be converted from strings to ints
6  """
7  # Constant for filename
8  FILE_NAME = 'numbers.txt'
9
10
11 def main():
12
13     # Catch any exceptions
14     try:
15         # Open a file for reading
16         with open(FILE_NAME, 'r') as number_file:
17
18             # Read 3 numbers from a file
19             number1 = int(number_file.readline())
20             number2 = int(number_file.readline())
21             number3 = int(number_file.readline())
22
23             # Sum the numbers
24             total = number1 + number2 + number3
25
26             # Display the numbers and the total
27             # A different way of printing numbers as strings
28             print(f'The numbers are: {number1} {number2} {number3}')
29             print(f'The total is: {total}')
30
31         # Let the user know there was trouble
32     except:
33         print('There was trouble reading the file.')
34
35
36 # Call the main function.
37 if __name__ == '__main__':
38     main()
```

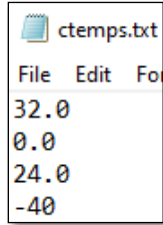
Example run:

```
The numbers are: 3 4 5
The total is: 12
```

Tutorial 8.7 – Reading and Writing to a Text File

We will write a program that reads a list of temperatures from a file called **ctemps.txt**, converts those temperatures to Fahrenheit, and writes the results to a file called **ftemps.txt**.

Create a text file named **ctemps.txt** in the current folder and put the following numbers in it. Don't put any extra line returns in it, stop at typing at -40.



Create and test the following program named **convert_temperatures.py**

```

1  '''
2      Name: convert_temperatures.py
3      Author:
4      Created:
5      Purpose: Read a list of temperatures from a file, convert them to Fahrenheit
6      Write the results to a file and to the screen
7  '''
8  # Constant for filename
9  CFILE = 'ctemps.txt'
10 FFILE = 'ftemps.txt'
11
12 def main():
13
14     # Catch any exceptions
15     try:
16         # Open a file for writing
17         cfile = open( CFILE, 'r')
18         ffile = open( FFILE, 'w')
19
20         # Strip the newline character off of every line in the file
21         ctemp = [line.strip() for line in cfile ]
22
23         # Loop until all the temperatures have been converted
24         for t in ctemp:
25             # Convert the temperature to Fahrenheit
26             ftemp = float(t) * (9.0 / 5.0) + 32.0
27             # Write the temperature to the file
28             ffile.write( f'{ftemp}\n' )
29             # Print the conversion to the screen
30             print(f'{t} Celcius equals {ftemp} Fahrenheit')
31
32         # Let the user know there was an exception
33     except:
34         print('A file exception occurred.')
35
36     # Regardless of what happens, the file is closed
37     finally:
38         # Close both files
39         cfile.close()
40         ffile.close()
41
42     # Call the main function
43 if __name__ == '__main__':
44     main()

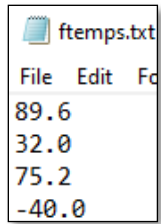
```

Example run:

```

32.0 Celcius equals 89.6 Fahrenheit
0.0 Celcius equals 32.0 Fahrenheit
24.0 Celcius equals 75.2 Fahrenheit
-40 Celcius equals -40.0 Fahrenheit

```



Assignment Submission

1. Attach the pseudocode.
2. Attach the program files.
3. Attach screenshots showing the successful operation of the program.
4. Submit in Blackboard.